Input : An Array of integers $A[0 \dots n-1]$

Output: False, if there are duplicates. True, otherwise

```
for $i \leftarrow 0$ to $n-2$ do
    for $j \leftarrow i+2$ to $n-1$ do
        if $A[i] = A[j]$ then
            return False
return True
```

- Time Complexity?

Input : Two square matrices A, B of dimension n

Output: A matrix $C = AB$

```
for $i \leftarrow 0$ to $n - 1$ do
    for $j \leftarrow 0$ to $n - 1$ do
        $C[i, j] \leftarrow 0$
        for $k \leftarrow 0$ to $n - 1$ do
            $C[i, j] \leftarrow C[i, j] + A[i, k] * B[k, j]$
return $C$
```

- Time Complexity?

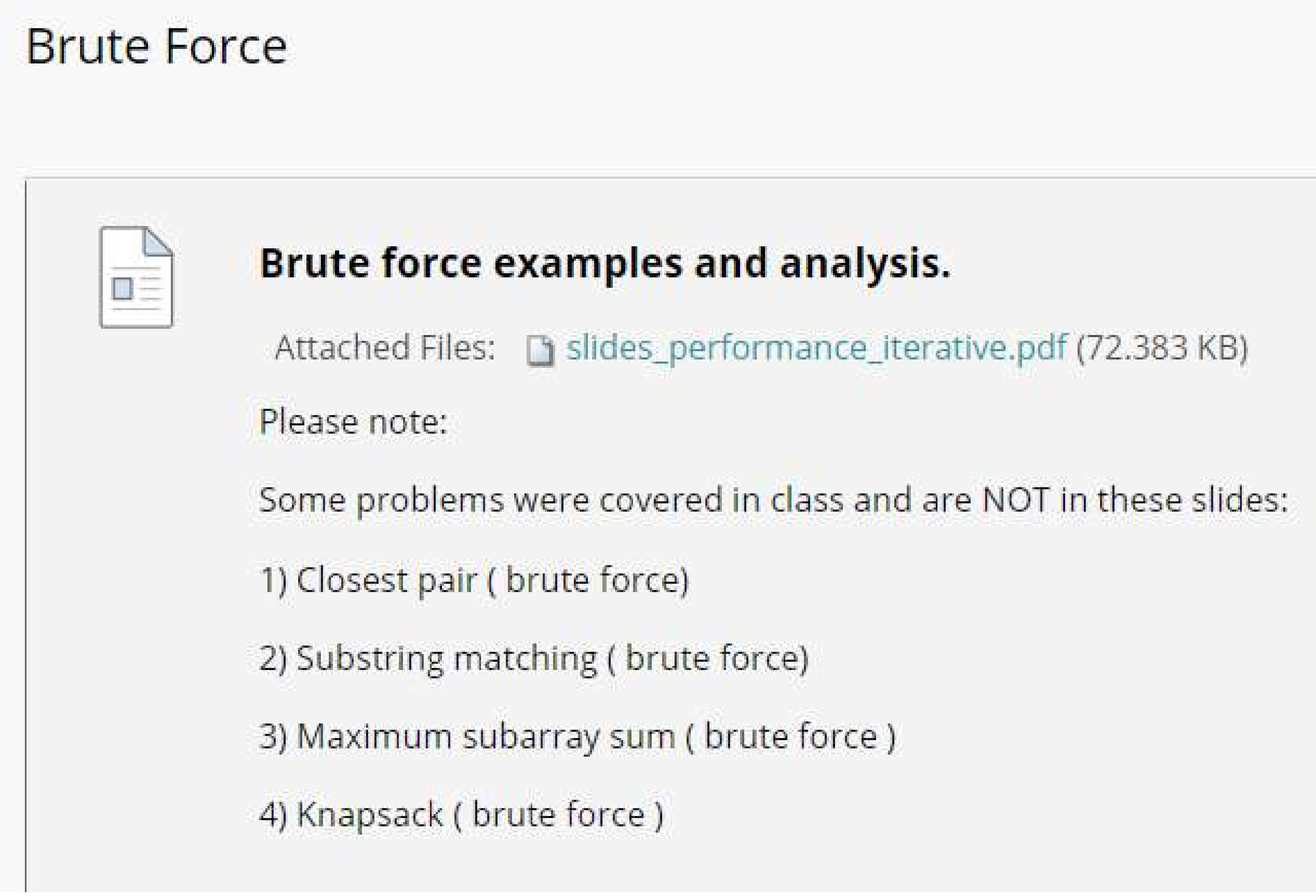
Input : Sorted array of integers $A[0 \ldots (n-1)]$ and a target integer x

Output: If x is in A, return its index. Returns -1 otherwise

```
$low \leftarrow 0$
$high \leftarrow n - 1$
while $low \leq high$ do
    $mid \leftarrow (low + high)/2$
    if $A[mid] = x$ then
        return $mid$
    if $A[mid] > x$ then
        $high \leftarrow mid - 1$
    else
        $low \leftarrow mid + 1$
return $-1$
```

- Time Complexity?

```
Input  : An integer $n \geq 3$
Output: True if $n$ is prime, False otherwise
for $i \leftarrow 2$ to $n - 1$ do
    if $i$ divides $n$ then
        return False
return True
```



Brute Force

Brute force examples and analysis.

Attached Files: slides_performance_iterative.pdf (72.383 KB)

Please note:

Some problems were covered in class and are NOT in these slides:

1) Closest pair (brute force)

2) Substring matching (brute force)

3) Maximum subarray sum (brute force)

4) Knapsack (brute force)